

Tópicos Abordados



- Conceitos de C# usados pelo LINQ
- Funcionamento do LINQ
- Deferred e immediate execution
- A classe *System.Ling.Enumerable*
- Método OfType<T>()
- Projetando tipos de dados anônimos
- Operações com conjuntos
- · Agrupamento de dados
- Operador join
- Geração de dados

LINO



- Permite a construção de queries para extrair dados
 - O LINQ tem algumas similaridades com a linguagem SQL
- Extração de dados de várias fontes
 - Objetos (coleções e arrays)
 - -XML
 - DataSet
 - Entities

Conceitos Necessários



- O uso de LINQ se apoia em alguns conceitos da linguagem C#
 - Expressões lambda
 - Extension methods
 - Tipo de dados *var*
 - Tipos anônimos
- Os dois últimos serão abordados a partir de agora

O Tipo *var*



- Variáveis locais em C# podem ser declaradas usando o tipo var
 - O tipo de dado da variável é definido de forma implícita pelo compilador

 Depois que o tipo da variável é definido, não pode ser trocado



Não compila

Regras de Uso do *var*



- Uso restrito a variáveis locais (definidas dentro de método ou properties)
 - Parâmetros de métodos, retorno ou fields não podem usar o *var*
- A variável deve receber um valor no momento da declaração
 - É através do valor que o compilador vai decidir o tipo de dado de forma implícita
 - A primeira atribuição não pode ser *null*

Tipos Anônimos



- Um tipo anônimo é uma classe criada sem nome
 - O nome é gerado apenas no momento da compilação
 - O nome não precisa ser conhecido
- Tipos anônimos são bastante úteis em situações onde você deseja criar classes que existem apenas para encapsular dados e serão usadas apenas em locais específicos do código

Definindo Tipos Anônimos



- A variável que vai referenciar o objeto do tipo anônimo deve ser do tipo *var*
 - O tipo não é conhecido durante a escrita do código. Só será conhecido na compilação

var cachorro = new { Nome = "Totó", Idade = 5 };

Variável que referencia um objeto da classe anônima

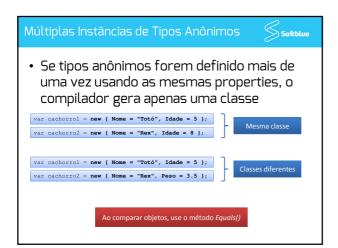
O Tipo Anônimo Internamente



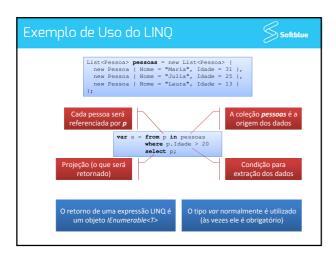
• Como a classe é definida internamente?

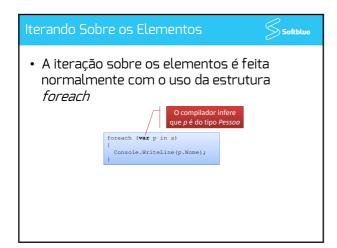


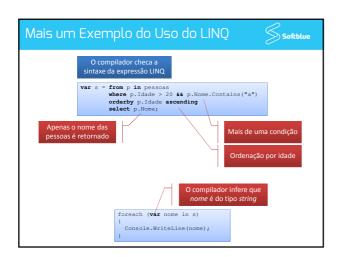


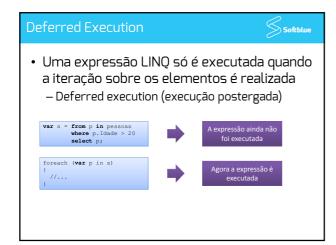


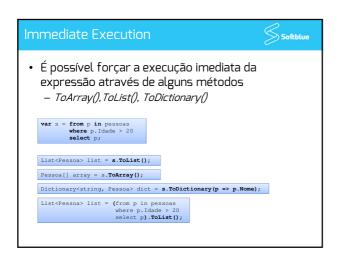












Para utilizar o LINQ para objetos, o assembly *System.Core.dll* deve ser referenciado Projetos do Visual Studio fazem esta referência de forma automática Todos os arquivos que usam expressões LINQ devem importar o namespace *System.Linq*

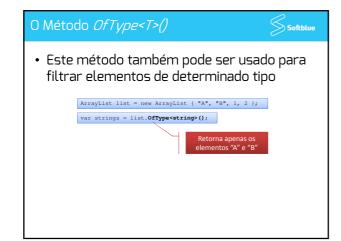
A Classe System.Ling.Enumerable

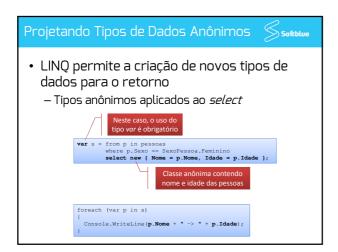


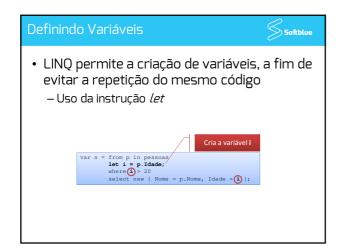
- Esta classe tem um papel fundamental na estrutura de funcionamento do LINQ
 - Ela adiciona uma série de extension methods à interface *IEnumerable<T>*
- Quando uma expressão LINQ é compilada, ela é transformada em chamadas a estes extension methods

A Classe System.Ling.Enumerable var s = from p in pessoas where p.Idade > 20 orderby p.Nome descending select p; var s = pessoas .Where(p => p.Idade > 20) .OrderByDescending(p => p.Nome) .Select (p => p)? Métodos como Where(), OrderByDescending(), Select(), etc. são definidos em Enumerable como sendo extension methods de lEnumerable<T>

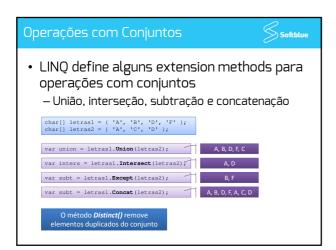
LINQ não consegue trabalhar com coleções que não usam generics | ArrayList pessoas = new ArrayList { | new Pessoa { Nome = "Maria", Idade = 31 }, new Pessoa { Nome = "Julia", Idade = 25 }, new Pessoa { Nome = "Laura", Idade = 13 } } | A classe System.Linq.Enumerable define um método chamado OfType<T>() | Converte para um tipo genérico | Var s = from p in pessoas.ofType<Pessoa>() | Transforma um | Enumerable em um | Enumerable em um | Enumerable | Enu

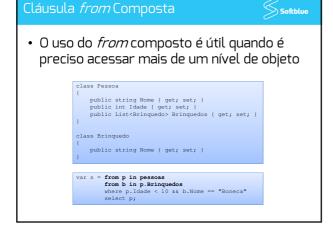


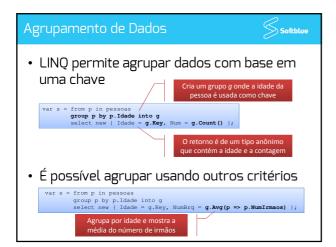


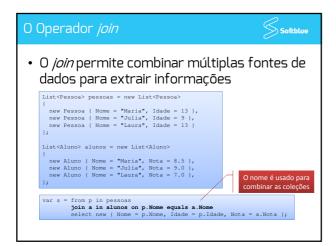


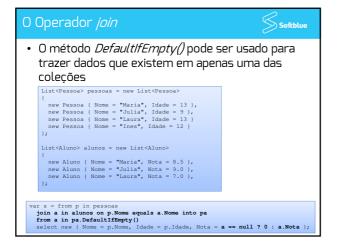
• LINQ possui operações que agregam os elementos da coleção • Só podem ser chamados como métodos — Extension methods de *IEnumerable<T>*double {| notas = { 9.5, 7.0, 6.5, 8.0 }; int count = notas.Min(); double min = notas.Min(); double way = notas.Max(); Média double sum = notas.Sum(); Soma











• LINQ também possui métodos para gerar dados – Métodos estáticos na classe *Enumerable* • Range() • Empty() • Repeat() Geração de Dados – *Range()* • Gera um intervalo numérico – Apenas números inteiros var v = Enumerable.Range(1, 5); 1,2,3,4,5 • O método *Select()* pode ser usado para modificar o comportamento do intervalo var v = Enumerable.Range(1, 5).**Select**(n => n * 2); Softblue • Gera uma coleção vazia – Útil em situações onde é necessário fornecer como parâmetro ou retornar uma coleção vazia var v = Enumerable.Empty<int>();

